

ADA 036868

RADC-TR-77-13  
Final Technical Report  
January 1977



SYSTEM MODULARIZATION TO MINIMIZE LIFE CYCLE COSTS

Syracuse University

COPY AVAILABLE TO DDC DOES NOT  
PERMIT FULLY LEGIBLE PRODUCTION

Approved for public release;  
distribution unlimited.

Copy available to DDC does not  
permit fully legible reproduction



ROME AIR DEVELOPMENT CENTER  
AIR FORCE SYSTEMS COMMAND  
GRIFBS AIR FORCE BASE, NEW YORK 13441



This report contains a large percentage of machine-produced copy which is not of the highest printing quality but because of economical consideration, it was determined in the best interest of the government that they be used in this publication.

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public including foreign nations.

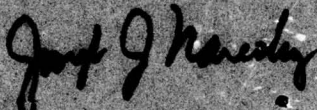
This report has been reviewed and is approved for publication.

APPROVED:



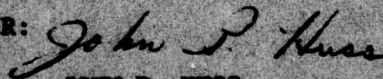
ANTHONY COPPOLA  
Project Engineer

APPROVED:



JOSEPH J. NARESKY  
Chief, Reliability and Compatibility Division

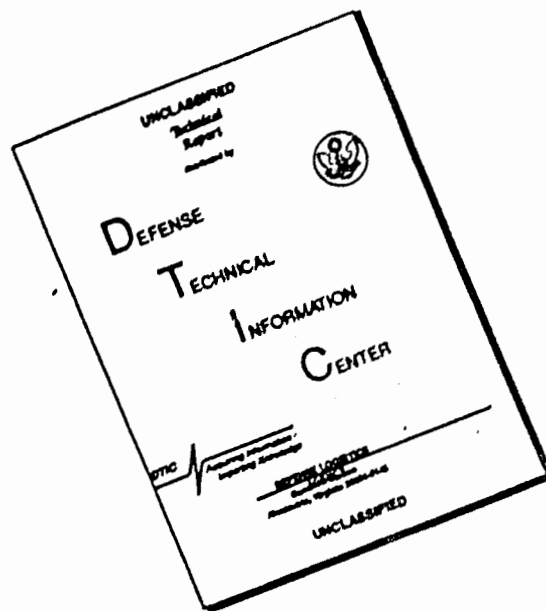
FOR THE COMMANDER:



JOHN P. HUSS  
Acting Chief, Plans Office

Do not return this copy. Retain or destroy.

# DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

UNCLASSIFIED  
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-77-13 (19)	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) SYSTEM MODULARIZATION TO MINIMIZE LIFE CYCLE COSTS	5. TYPE OF REPORT & PERIOD COVERED (9) Final Technical Report, Aug 75 - Jul 76	
7. AUTHOR(s) (10) John E. Biegel Bisrat/Bulcha	6. PERFORMING ORG. REPORT NUMBER N/A	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Syracuse University Skytop Office Bldg. Syracuse NY 13210	8. CONTRACT OR GRANT NUMBER(s) (15) F30602-71-C-0312	
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (RBRT) Griffiss AFB NY 13441	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 62702F 45400526	
12. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Same (16) 4540 (17) 05	12. REPORT DATE (11) January 1977	
	13. NUMBER OF PAGES 54 (12) 51p	
	15. SECURITY CLASS. (of this report) UNCLASSIFIED	
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) Same		
18. SUPPLEMENTARY NOTES RADC Project Engineer: Anthony Coppola (RBRT)		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Life Cycle Costs System Design System Partitioning Modules		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) When designing equipments, particularly electronic equipments, there is a need to collect components into integral groups called modules. A network can be arbitrarily decomposed into modules, but an optimum modularization would minimize the life cycle cost (LCC) of the system. This research was directed toward that goal.  The method developed is a heuristic and no attempt has been made to establish		



**UNCLASSIFIED**

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

its optimality (although no counter examples are known). The problem is approached as one of decomposing a network subject to constraints. The problem may be stated as:

Decompose a system to minimize the life cycle cost subject to physical constraints, a mean-time-to-repair constraint and establish spares to meet an availability constraint.

A computer routine has been developed and explained. A flow chart is included.

**UNCLASSIFIED**

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

## EVALUATION

This study was one of many responsive to Technical Planning Objective #13 (Reliability) performed by Syracuse University under Contract No. F30602-71-C-0312. The objective of this study was the development of mathematical models for selecting a set of modules which represents the division of an electronic system resulting in the lowest life cycle costs. The study provided a method based on network theory which, while not proven to provide the optimal solution, will certainly provide an efficient means for reducing system life cycle costs through better partitioning in design.

Future work related to this effort will be the development of other tools for lowering life cycle costs through more cost-effective design, and the incorporation of these into Air Force standards and handbooks to assure their application.

*Anthony Coppola*

ANTHONY COPPOLA  
Project Engineer

ACCESSION for	
NTIS	White Section <input checked="" type="checkbox"/>
DDC	Buff Section <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
JUSTIFICATION	
BY	
DISTRIBUTION/AVAILABILITY CODES	
Dist.	AVAIL. and SPECIAL
A	



## I. INTRODUCTION

The objective of this research was to develop a method of establishing modules for equipments in a way that would "minimize" the life cycle cost of the system. The construction of the modules was to be as non-restrictive as possible without an exhaustive enumeration of all possible modules. (Total or exhaustive enumeration, while feasible for small networks would be infeasible for large networks. A network of ten nodes may contain more than 115,000 different modules.) For all practical purposes, the number of possible modules in a 100 node network is infinite. Jensen (1,2), Caponecchi and Jensen (3) and Caponecchi (4) have proposed the use of proper, restricted cuts as a basis for establishing modules. In this work, those restrictions have been removed. Proper, restricted cuts are too restrictive and could result in the construction of modules that are extremely non-optimal.

The solution procedure developed in this research requires that the network be partitioned  $N$  times, where  $N$  is the number of nodes in the network. The "best" of the  $N$  modular arrangements is chosen as the desired modular structure. The "best" structure is that structure which has the lowest life cycle cost. A computer routine and numerical example are given.

## II. NETWORK STRUCTURE

The systems under consideration must be capable of being structured as networks. (Electronic networks are prime examples.) Each node in the network must consist of an element that cannot feasibly be decomposed. In other words, each node can be called a "smallest functional element"; that is an SFE. The arcs in the network represent the interconnections

between nodes or SFE's. In essence, they (the arcs) are distinct, discrete channels by which the nodes communicate. An example network is shown in Figure 1.

Networks can be classified in several ways. They are graphs. In fact, they are geometric graphs and may be planar or non-planar, directed or non-directed and feed-forward or feedback. The developments that follow apply to any network that fits into the above description. (For a more detailed description of graphs, see Busacker and Saaty, Finite Graphs and Networks, McGraw-Hill, 1965, Chapters 1 and 2.)

### III. NETWORK PARTITIONING

Networks can be divided into subnetworks or modules by two basic types of partitioning. The two types are 1) those minimum partitions that disrupt all communication paths from source-to-sink, and 2) those partitions that do not disrupt all communication paths from source-to-sink. The network in Figure 2a is partitioned to disrupt all paths while the same network shown in Figure 2b is partitioned so that not all paths are disrupted.

The first type of partitioning is called partitioning by proper cuts. When the network has directed arcs, it is possible to use restricted proper cuts. The method being proposed permits any combination of both types of partitioning. The partitioning may result in disjoint elements (those with no direct interconnections) being assigned to a module. An example of such partitioning is shown in Figure 3.



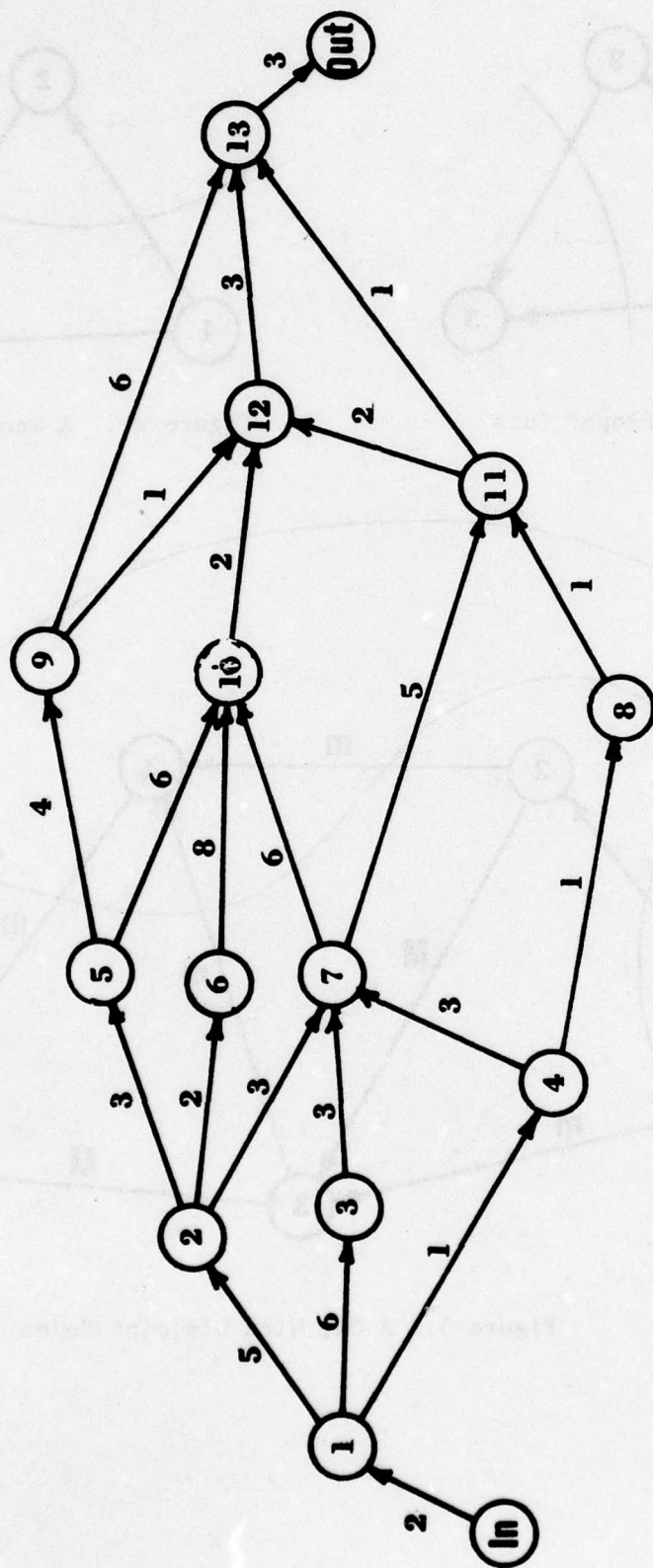


Figure 1. Example Network--A Functional Schematic (After Jensen (1))

Note: Arc labels are number of interconnections between nodes.

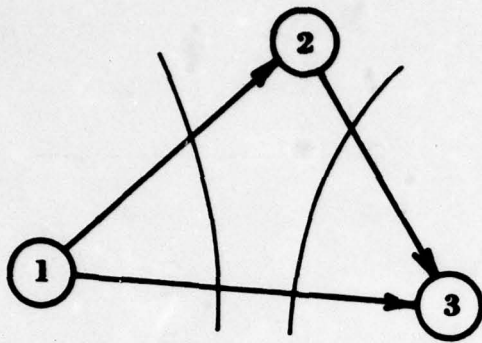


Figure 2a. Two Proper Cuts

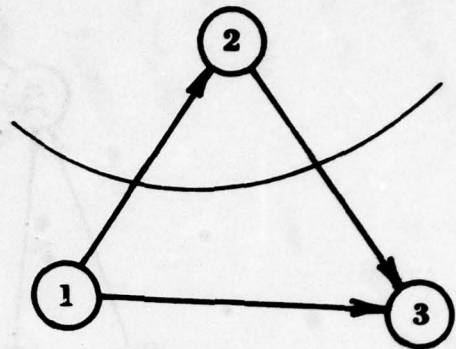


Figure 2b. A Generalized Cut

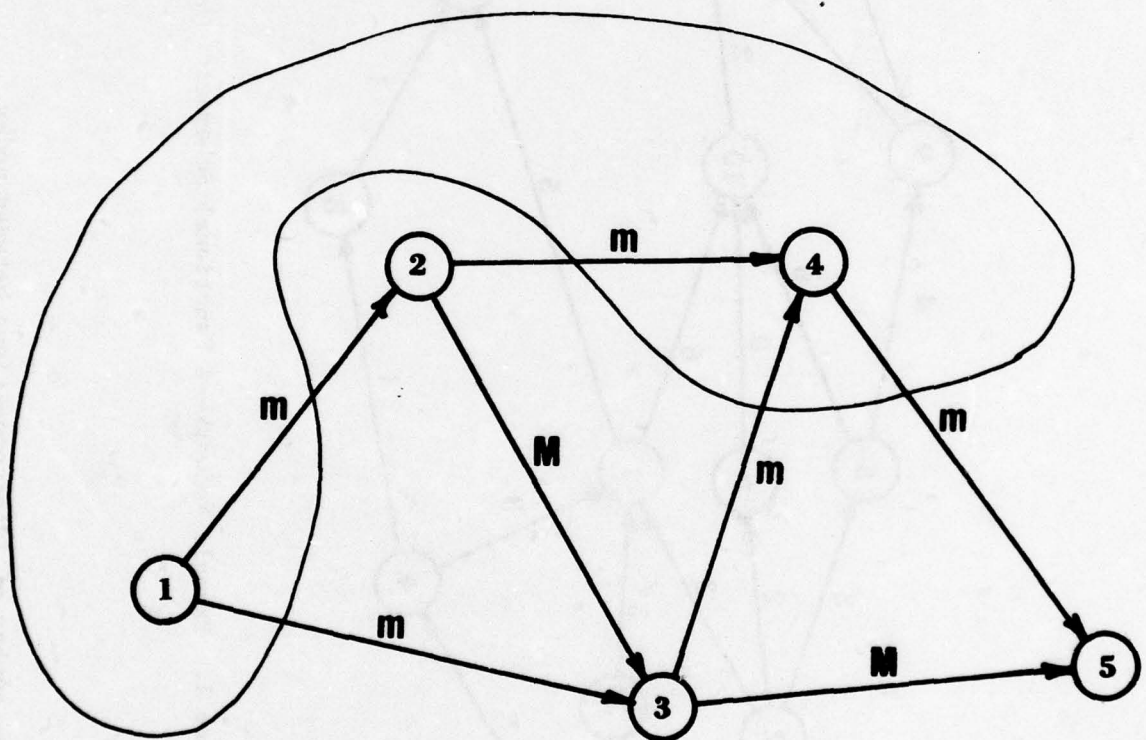


Figure 3. A Cut With Disjoint Nodes



#### IV. PROPER CUT PARTITIONING

Given a linear graph (network) a proper cut is defined as a set of arcs which when removed from the graph divide it into exactly two connected subgraphs. For a graph of  $N$  nodes there are a maximum of  $2^N$  different partitions of the nodes into two subsets  $X$  and  $\bar{X}$  where  $X$  and  $\bar{X}$  include all nodes. When two nodes are specified (such as an input and an output node) the number of proper cuts is reduced to  $2^{N-2}$ . Jensen (2) presents an algorithm that uses a tree search approach for determining proper cuts. The generation of proper cuts is a lengthy one and Jensen proposes the use of restricted proper cuts to decompose the graph. A restricted proper cut is defined as a set of nodes  $X$  such that if there is an arc from node  $i$  to node  $j$ , and if node  $j$  belongs to  $X$ , then node  $i$  must also belong to  $X$ .

The network of Figure 1 has 187 proper cuts and 69 restricted proper cuts. A network with thirteen nodes could have as many as  $2^{13}$  or 8192 proper cuts. The network can be divided into 1533 modules by using proper cuts and only 696 modules by using restricted proper cuts. There could be as many as  $\sum_{n=1}^{13} S(13,n)$  cuts where  $S(13,n)$  is the Sterling number of the second kind ( $S(m,n) = \sum_{i=0}^n (-1)^i \binom{n}{i} (n-i)^m / n!$ ).

Suppose the network in Figure 4a must be partitioned. Nodes 1, 2, and 3 cannot be in a single module; however, any pair of nodes can be in the same module. Let node 1 be the source node and node 3 be the sink node. The penalty for cutting arc (1,3) is very high, say  $M$ , whereas the penalty for cutting arcs (1,2) and (2,3) is very small, say  $m$ , where  $2m \ll M$ . Decomposition by proper cuts would force arc (1,3) to be cut and, therefore, experience the high penalty ( $M+m$ ). The network would be optimally

decomposed by making two sets of nodes, set 1 contains node 1 and node 3 and set 2 contains node 2 at a small penalty ( $2m$ ), which is very much less than the penalty for cutting arc (1,3). This partitioning by proper cuts is shown in Figures 4a and 4b. The optimal partitioning is shown in Figure 4c.

#### V. LIFE CYCLE COST MODEL

Life Cycle Cost (LCC), as used in this work, is the cost of acquisition plus the support cost for an equipment over its intended useful life. The LCC model used is after Caponecchi (4) and is:

$$LCC = Ct(M_1) + Ct(M_2) + \dots + Ct(M_i) + \dots + Ct(M_n),$$

where LCC = Life Cycle Cost

$M_i$  = Module  $i$  of design  $m$

$Ct(M_i)$  = The cost of acquiring and maintaining  
module  $i$  generated by design  $m$ .

$n$  = Total number of modules in design  $m$ .

$$\text{or } LCC = \sum_{i=1}^n Ct(M_i). \quad (1)$$

Since the Life Cycle Cost is composed of two major elements; namely, the acquisition cost and the life time support cost; the acquisition cost (not including spares) can be expressed in terms of the candidate modules. That expression is:

$$Ca = Ne \sum_{i=1}^n C(M_i) \quad (2)$$

Where  $Ca$  = Acquisition Cost for  $Ne$  equipments

$Ne$  = Number of equipments to be procured

$C(M_i)$  = Cost of acquiring module  $i$  of design  $m$ .



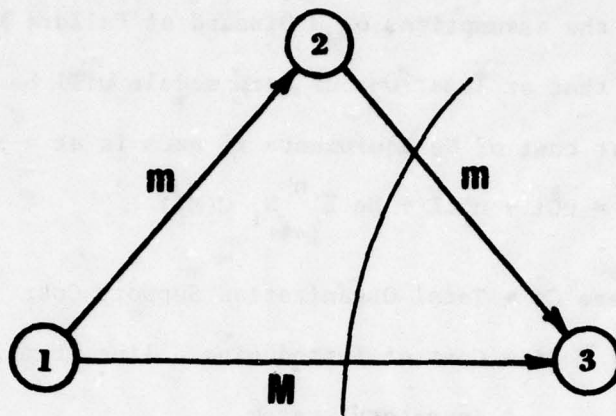


Figure 4a. A Proper Cut at Penalty  $m+M$

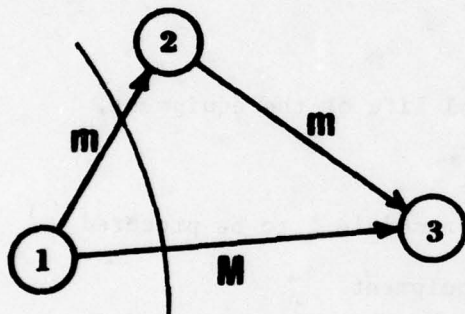


Figure 4b. Another Proper Cut at Penalty  $m+M$

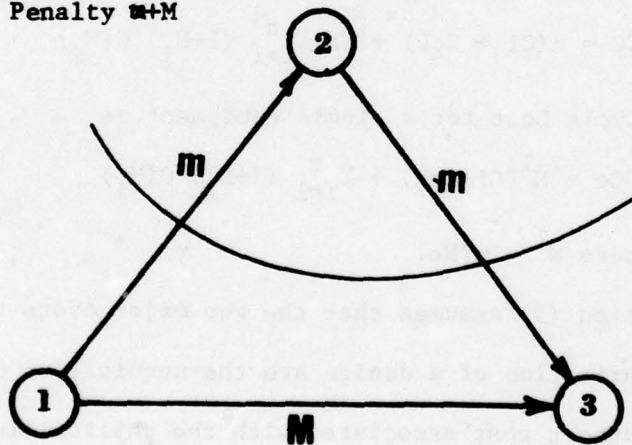


Figure 4c. The Optimal Cut--Not a Proper Cut

Under the assumptions of a Discard at Failure Maintenance (DAFM) policy and that at least one of each module will be spared, the life time support cost of Ne equipments if each is at a separate site is:

$$Cs = nCi + nCcL + Ne \sum_{i=1}^n N_i C(M_i) \quad (3)$$

Where Cs = Total Organization Support Cost

Ci = Cost of introducing a line item into the inventory system

Cc = Cost of maintaining a line item in inventory for one year

L = Planned operational life of the equipment, in years

$N_i$  = Number of spares of module i to be procured to support each equipment

The total Life Cycle Cost is the sum of Equations (2) and (3), or:

$$LCC = n(Ci + CcL) + Ne \sum_{i=1}^n (1+N_i) C(M_i) \quad (4)$$

The Life Cycle Cost for a single equipment is

$$LCCe = N^1(Ci+CcL) + \sum_{i=1}^n (1+N_i) C(M_i) \quad (5)$$

Where  $N^1 = n/Ne$ .

Equation (5) assumes that the two major costs that are affected by the modularization of a design are the acquisition cost and the organization support cost associated with the physical inventory of parts. It ignores the cost of maintenance personnel and consumables on the assumption that they will be relatively unaffected by the modularization decisions. It further assumes there will be "logistic self-sufficiency". The implication being that the shelf life of all items in the spares inventory exceeds the planned life of the equipment and that there is no



deterioration during non-use. A further assumption is that there will be only one level of modularization.

NOTE: If Equation (4) is minimized on the basis of an integer number of spares for each equipment,  $N_1$  is integer. If the number of spares for all equipments is to be integer, then  $NeN_1$  must be integer.

#### VI. ADEQUACY OF SPARES CONSTRAINT

All modules are assumed to have a constant failure rate while in service and a zero failure rate while in spares. With these assumptions, the probability that  $N_1$  spares will be adequate to meet the availability over the planned life,  $L$ , of the equipment is:

$$P(w_1 \leq N_1) = \sum_{w=0}^n \frac{e^{-f_1} f_1^{w_1}}{w_1!} \geq A \quad (6)$$

where  $P(w_1 \leq N_1)$  = Probability that the number of failures of module  $i$  is no larger than the number of spares of module  $i$ .

$w_1$  = Number of failures of module  $i$  in  $L$ .

$f_1$  = Expected number of failures of module  $i$  in  $L$ .

$A$  = The minimum allowable availability of an equipment.

An expression for  $f_1$  is:

$$f_1 = r_1 L Ne, \quad (7)$$

where  $r_1$  = failure rate of module  $i$ .

The failure rate of module  $i$ ,  $r_1$ , is sensitive to the modularization since the failure rate of a module is a function of the failure rates of the

elements in the module and the number of external and internal connections.

This can be expressed as:

$$r_i = \sum_{j \in M_i} r_j - \Delta P(M_i) r_p, \quad (8)$$

where  $r_j$  = The failure rate of element  $j$  of module  $i$

$\Delta P(M_i)$  = The number of external connections eliminated by putting elements  $j$  in module  $i$ .

$r_p$  = Failure rate correction factor for interconnection reduction.

An important point to be noted is that the value of  $r_i$  in Equation (8) is linearly dependent on the number of external connections eliminated by the modularization.

## VII. PHYSICAL CONSTRAINTS

In the modularization of equipments, there will be some physical constraints which cannot be violated. In electronics modularization, the constraints might be: weight, heat generated, external connections, number of elements, chip area, power consumption, physical and electronic compatibility, etc. The exact nature of the constraints is not as important as their form. For example, linear constraints are usually much easier to handle than non-linear constraints. However, non-linear constraints that are convex will usually give less trouble than non-linear constraints that are neither concave nor convex. Caponecchi (4) has assumed some linear constraints which do not cause any difficulty. Such physical constraints as are mentioned above can be expressed as:

$$\sum_{j \in M_i} A_{jk} \leq A_{ik}^{\max} \quad (9)$$

where  $A_{jk}$  = The value of characteristic k for element j  
assigned to module i.

$A_{ik}^{max}$  = Maximum value of characteristic k that may be  
assigned to module i.

### VIII. MAINTENANCE CONSTRAINT

The expected maintenance time for a module will be a function of the size of the module (the number of elements (SFE's) in the module) and the number of external connections to the module. Caponecchi (4) proposed that the maintenance time is composed of the sum of a constant time element, a time element that is a linear function of the number of modules in the equipment and a third time element that is exponentially related to the number of external connections to the module. A general expression for the expected maintenance time is:

$$E(TM) = T_1 + nT_2 + \sum_{i=1}^n T_3 \exp(T_4 P(M_i)) \quad (10)$$

where  $E(TM)$  = Expected maintenance time.

$T_1$  = A constant time per maintenance action.

$T_2$  = A constant time per module.

$T_3$  = A constant modifying the exponential relationship  
of the number of external connections, and

$T_4$  = A constant modifying the number of external connections.

$P(M_i)$  = Number of external connections to module i.

The specific form of Equation (10) used by Caponecchi (4) is:

$$E(TM) = 2.5 + 0.05n + 0.087 \sum_{i=1}^n \exp(0.047 P(M_i)). \quad (11)$$

This expression yields a constraint on the expected time for maintenance which must be less than the maximum mean time to repair; or

$$E(TM) \leq MTTR_{max}. \quad (12)$$

where  $MTTR_{max}$  = maximum allowable mean time to repair.



# IX. MATHEMATICAL FORMULATION OF THE MODULARIZATION PROBLEM

From the previous discussion, it is now possible to formulate the modularization problem. That formulation over all equipments is:

$$\text{Min. LCC} = n(C_i + C_c L) + N e \sum_{i=1}^n (1 + N_i) C(M_i)$$

Subject to:

$$\begin{aligned} 1) \quad & \sum_{i=1}^n \frac{e^{-f_i} f_i^{w_i}}{w_i!} \geq A \\ 2) \quad & T_1 + nT_2 + T_3 \sum_{i=1}^n \exp(T_4 P(M_i)) \leq \text{MTTRmax} \quad (13) \\ 3) \quad & \sum_{j \in M_i} A_{jk} \leq A_{i-k} \text{ max for all } i \text{ and } k \\ & N_i > 0 \text{ and integer.} \end{aligned}$$

The formulation for each equipment is:

$$\text{Min LCC} = N^1(C_i + C_c L) + \sum_{i=1}^n (1 + N_i) C(M_i). \quad (14)$$

This expression for the LCC is subject to the same constraints as listed above for Equation (13).

# X. DEPENDENCE OF LCC ON EXTERNAL CONNECTIONS

The mathematical formulation of the modularization problem shows that the solution is dependent on the number of external connections in three ways. First, the minimum LCC (objective function) is dependent on the number of external connections through  $C(M_i)$ , the cost of a module. Since internal connections are less expensive than external connections, the cost of modularizing  $N$  elements into modules will be reduced as some function of the reduction of external connections. The cost of packaging is also reduced. Second, the failure rate of a module is a function of the number of external connections as shown by Equation (8). The availability constraint is a function of the failure rate of the modules. And, third, the maintenance constraint is a function of the number of external

connections through its exponential term. Others have found the LCC to be highly dependent on the reliability. Hardy states "...Since hardware failure frequency is one of the key parameters that influences life-cycle cost..."\*

A further consideration is that there is no guarantee that there exists a modular arrangement that satisfies all constraints. Since the physical constraints restrict the size of modules and availability will increase with the size of the modules, the availability constraint may be set at a value that is unattainable in any modularization process. The same may be true for the mean-time-to-repair constraint.

#### XI. A MODIFIED PROBLEM

Since, as was shown in the previous section, the Life Cycle Cost is a function of the number of external connections between modules, the partitioning or modularization method will be based on the reduction in external connections achieved by the modularization. Assume the expected cost of an external connection is  $C_e$  and the expected cost of an internal connection is  $C_i$  where  $C_e$  is greater than  $C_i$ . The modified objective function is to minimize the cost for connections; that is:

$$\text{Min } C = \sum_{i=1}^n (E_i C_e + I_i C_i) \quad (15)$$

---

\*Hardy, C.A. "Avionic Reliability and Life-Cycle-Cost Relationship", AGARD Lecture Series No. 81 on Avionics Design for Reliability. (Available from NASA, Langley Field, Va. 23365, Attn: Report Distribution and Storage Unit.)

Where  $C$  = Cost of connections to and within a module,

$E_i$  = Number of external connections from module  $i$ ,

$C_e$  = Cost of an external connection to a module,

$I_i$  = Number of internal connections in module  $i$ , and

$C_i$  = Cost of an internal connection in a module.

The constraints on the physical design parameters will remain the same. The availability of the equipment is increased as the reliability is increased and the mean-time-to-repair is decreased as the number of modules and external connections is decreased. Thus, if the size of a module is at its maximum value, its availability will tend toward a maximum and its MTTR will tend toward a minimum.

The modularization process proceeds to combine SFE's in a sequential manner until no other SFE can be added without violating a constraint. The module so constructed is set aside and a new module constructed from the remaining SFE's. The process continues until all SFE's have been assigned to a module. Since the assignment of elements to modules under this procedure will be dependent on the starting point, it was decided to do  $N$  iterations with one iteration starting from each node. The best modular arrangement is then chosen from the  $N$  arrangements. (It is expected that the  $N$  arrangements are not unique so that fewer than  $N$  modular arrangements need to be evaluated.) The best module being the one with the smallest difference between external connections and internal connections. (This number can be negative.)

Once the best module is chosen, it must be evaluated for its mean-time-to-repair. If it does not meet the mean-time-to-repair constraint, the second best module is evaluated, then the third best, etc. until a



modular arrangement is found that satisfies the constraints. If no arrangement is found, the constraints are infeasible and either the design or the constraints must be revised to produce a feasible design.

## XII. PROCEDURE TO FIND MINIMALLY CONNECTED MODULES

Assume the network has been designed and the SFE's have been determined. The nodes (SFE's) should be consecutively numbered, preferably but not necessarily with the input node being node 1 and the output node being labelled N. It would be desirable to number the nodes such that arc (i,j) always proceeds from node i to node j where  $j > i$ .

After the nodes have been numbered, proceed as follows:

0. Put  $k = 1$
1. Put all nodes in set MN
2. Put  $MN' = MN$
3. Put  $j = 1$
4. Put smallest numbered node, not assigned to a module, into module  $M_j$ . Remove the node from  $MN'$ .
5. Start with the smallest numbered node in  $MN'$ . Form pairs between all nodes in  $M_j$  collectively and all nodes in  $MN'$ . For each combination, calculate:  
$$EXT = \text{Number of external connections}$$
$$INT = \text{Number of internal connections}$$
$$ST = EXT - INT$$
6. Select the combination for which ST is minimum (use first minimum if more than one exists).
7. Check for conformance and to physical constraints. If all constraints are satisfied, replace  $M_j$  with new combination and to to Step 9. Otherwise to to Step 8.

8. Put  $M_j$  aside, put  $j = j + 1$ .
9. Remove the node most recently added to  $M_j$  from  $MN'$ . If  $MN' - M_j$  is null, go to Step 10. Otherwise go to Step 5.
10. After all nodes have been assigned to modules, store the modules as set  $S_k$ . Put  $k = k + 1$ . Put  $i = i + 1$ . If  $i + 1 > N$ , go to Step 11. Otherwise go to Step 1.
11. Find that set of modules with the smallest value of value of  $ST$  when summed over all modules in the set.
12. Calculate the  $MTTR$ . If the  $MTTR \leq MTTR_{max}$ , STOP. This is the solution. Otherwise remove current set of modules from further consideration and go to Step 11. If no set of modules satisfies the  $MTTR$  constraint, design is infeasible.

### XIII. CALCULATION OF THE NUMBER OF EXTERNAL AND INTERNAL CONNECTIONS FOR THE MODULES

A network can be represented by a matrix and this is the representation used to calculate the values of  $EXT$ ,  $INT$  and  $ST$  in previous solution procedure. The matrix representation of the network in Figure 1 (without input and output) is shown in the matrix in Figure 5. The entries in row  $i$  are the connections out of  $i$  and entries in column  $j$  are the connections into  $j$ . The entries on the main diagonal are the total number of connections to node  $i$  (both into and out of node  $i$ ).

The determination of the best node to associate with node  $l$  is shown in Table 1.

	j												
	1	2	3	4	5	6	7	8	9	10	11	12	13
i	1	12	5	6	1								
2			13		3	2	3						
3				9			3						
4					5		3	1					
5						13			4	6			
6							10			8			
7								20		6	5		
8									2		1		
9										10		1	5
10											22	2	
11												10	2
12													8
13													

Figure 5: Matrix Representation of Network Shown in Figure 1

Table 1. Determination of Best Node to Associate With Node 1

Node Pair	External Connections (EXT)	Internal Connections (INT)	ST=EXT-INT
1,2	15	5	10
1,3	9	6	3 (Minimum)
1,4	15	1	14
1,5	25	0	25
1,6	22	0	22
1,7	32	0	32
1,8	14	0	14
1,9	22	0	22
1,10	34	0	34
1,11	21	0	21
1,12	20	0	20
1,13	21	0	21



Table 1 can be found as follows:

EXT = Sum of entries in row 1 (except for main diagonal element)  
+ sum of entries in row 2 (except for main diagonal elements)  
- the entry at 1, 2.

$$= (5+6+1) + (3+2+3) - 5 = 15$$

INT = Entry at intersection 1, 2

$$= 5$$

$$ST = EXT - INT = 15 - 5 = 10$$

The minimum difference between the number of external connections and the number of internal connections occurs when nodes 1 and 3 are paired. The pairing of nodes 1 and 3 must be checked against the constraints. The physical characteristics of the nodes, as used by Caponecchi, is given in Table 2.

The combination of nodes 1 and 3 has: 59 Part Count, 385 Heat Generation, .049 Required Chip Area and 9 External Connections. The maximum allowable values are 150,1100,0.17 and 25 respectively. So Nodes 1 and 3 should be combined into a module.

The matrix can be reconstructed by combining nodes 1 and 3 into node 1'. Two entries are made in the 1', 1' location the values are the total number of external connections/the number of internal connections. The matrix with this entry is shown in Figure 6. Using Figure 6, the best node to be combined with 1, 3 is 2. The value of ST for nodes 1, 3, 2 in a module is 1 (the external connections are  $(3+2+3+3+1)$  and the internal connections are  $(5+6)$ ). The computations are shown in Table 3.

Table 2. Physical Characteristics of the Nodes\*

Node	Part Count	Heat Generation	Required Chip Area	Number of External Connections	Failure Rate/Hr. $\lambda_1 \times 10^{-5}$
1	27	180	.024	12	.211
2	43	210	.026	13	.450
3	32	205	.025	9	.225
4	40	290	.038	5	.300
5	25	220	.020	13	.190
6	15	140	.022	10	.178
7	27	185	.024	20	.420
8	18	150	.020	2	.080
9	23	200	.022	10	.160
10	50	260	.050	22	.350
11	26	150	.023	9	.182
12	17	145	.020	8	.140
13	12	80	.015	10	.170

\*maximum  
allowable  
per module 150

1100

0.17

25

	j												
	1'	2	4	5	6	7	8	9	10	11	12	13	
1'	9/6	5	1										
2		13		3	2	3							
4			5			3	1						
5				13				4	6				
6					10				8				
7						20			6	5			
8							2			1			
9								10			1	5	
10									22		2		
11										9	2	1	
12											8	3	
13												9	

Figure 6: First Reduced Matrix

Table 3. Determination of Best Node to Associate With The Node 1,3 Combination

	External	Internal	ST	
132	12	11	1	Minimum ST
134	12	7	5	
135	22	6	16	
136	19	6	13	
137	23	9	14	
138	11	6	5	
139	19	6	13	
1310	31	6	25	
1311	18	6	12	
1312	17	6	11	
1313	18	6	12	



The physical characteristics are:

Part Count  $102 < 150$

Heat Generation  $595 < 1100$

Required Chip Area  $.075 < .17$

Number of External Leads  $12 < 25$ .

So a module can be made from nodes 1, 2 and 3.

Proceeding in the above manner, the best modules starting from node 1 are:

$M_1 = (1,2,3) \text{ (EXT=15, INT=12)}$

$M_2 = (5,6,9,10,12,13) \text{ (EXT= 14, INT=29) and}$

$M_3 = (7,8,11), \text{ (EXT=19, INT=6)}$

The total number of external connections is 48 and the number of internal connections is 47. The value of ST is 1.

The solution process now performs an additional N-1 iterations yielding N designs. These designs will probably not be unique, in fact, for the example problem, there are only 5 unique modularizations. These five designs are shown in Table 4. It should be noted that there are two designs with the same minimum number of external connections and that those modular arrangements are obtained by starting from any of the first 8 nodes. Under the assumed model, the mean-time-to-repair will be minimum when the number of external connections is minimum. It now becomes necessary to see if the MTTR of the designs created by the modularization process meet the MTTRmax requirement.

Table 4. Five Unique Designs

Starting Nodes	Elements in Modules	Total # External Connections
1,2,3	1,2,3,4	
	5,6,9,10,12,13	
	7,8,11	48
4,5,6,7,8	3,4,7,8,11	
	5,6,9,10,12,13	
	1,2	48
9	5,8,9,11,12,13	
	4,6,7,10	
	1,2,3	52
10	2,5,6,10	
	4,8,9,11,12,13	
	1,3,7	58
11 12 13	5,8,9,11,12,13	
	1,2,3,4,	
	6,7,10	56

#### XIV. CHECKING MTTR CONSTRAINTS

The model assumed that the expected time for maintenance took the form:

$$E(TM) = 2.5 + 0.05n + 0.087 \sum_{i=1}^n \exp(.047P(M_i)). \quad (11)$$

The MTTR will be checked for conformance to the MTTR constraint. The  $MTTR_{max}$  has been set at 3.36. The calculations are shown in Table 5.

Table 5. MTTR Calculations

Module	EXT	INT	ST	$\exp(.047P(M_i))$
(1,2,3,4)	15	12	3	2.0239
(5,6,9,10,12,13)	14	29	-15	1.9309
(7,8,11)	19	6	13	2.4572

$$E(TM) = 2.5 + 0.15 + 0.558 = 3.208 < 3.36.$$

Thus, from Table 5, it can be seen that the modularization does meet the MTTR constraint.

#### XV. SPARES REQUIREMENT TO SATISFY AVAILABILITY

Since every element must function for the system to function, the modules can be treated as a series system from the reliability standpoint. The availability constraint is dependent on the module reliability in accordance with Equations (6), (7), and (8). That is:

$$P(w_1 \leq N_1) = \sum_{w_1=0}^n \frac{e^{-f_1} f_1^{w_1}}{w_1!} \geq A \quad (6)$$

$$f_1 = r_1 L N e, \text{ and} \quad (7)$$

$$r_1 = \sum_{j \in M_1} r_j - \Delta P(M_1) r_p. \quad (8)$$

$$\text{Where } r_p = 1 \times 10^{-6}.$$



The cost of a module,  $C(M_1)$ , is:

$$C(M_1) = C_1 N_p + C_2 \text{ EXT} + C_3$$

Where

$C_1$  = Components parts cost factor

$N_p$  = Number of parts in the module

$C_2$  = Interconnection cost factor

EXT = Number of interconnections

$C_3$  = Packaging cost

The failure and cost data are summarized in Table 6. These data are for the design that starts at nodes 1, 2, or 3. The total cost for the design is \$1,106. This design has the least cost of any of the

Table 6. Failure and Cost Data For a Modular Design

Module	Failure Rate 10 <sup>6</sup> Hrs.	Expected Number of Failures	Cost of the Module
(1,2,3,4)	26.9	71.0	\$ 414
(5,6,9,10,12,13)	25.9	68.0	412
(7,8,11)	7.8	20.5	280

designs shown in Table 4.

The availability problem can be expressed as:

$$\text{Min } C = 414N_1 + 412N_2 + 280N_3$$

subject to:

$$(\sum_{x=0}^{N_1} (\exp(-71)) (71^x) / x!) \cdot (\sum_{x=0}^{N_2} (\exp(-68)) (68^x) / x!).$$

$$(\sum_{x=0}^{N_3} (\exp(37)) (37^x) / x!) \geq 85 \quad (16)$$

$$N_1, N_2, N_3 \geq 0 \text{ and integer}$$

Since the above problem is an integer non-linear programming problem, an approximate solution method will be used. The solution method is:

- 1) Initialize by finding a lower bound,  $N_j$ , for each module.

Choose  $N_j$  such that

$$\sum_{x=0}^{N_j} (\exp(-f_i)) (f_i^x) / x! \geq A = .85$$

- 2) Find the value of A for the system.

$$\text{Where } A = \prod_{i=1}^n (\sum_{x=0}^{N_i} (\exp(-f_i)) (f_i^x) / x!)$$

If  $A \geq .85$ , STOP.

- 3) Let  $N_i = N_i + 1$ .

$$4) \text{ Calculate } \Delta A_i = \left( \sum_{x=0}^{N_i+1} (\exp(-f_i)) f_i^x / x! \right) - \left( \sum_{x=0}^{N_i} (\exp(-f_i)) f_i^x / x! \right)$$

- 5) Calculate  $\Delta A_i / C(M_i)$

- 6) Choose  $\max_i \Delta A_i / C(M_i)$ . Set all other

$N_i = N_i - 1$ . Go to Step 2.

The calculations for the example are:

$$N_1 = 80, N_2 = 77, N_3 = 43.$$

And  $A = (.8692)(.8741)(.8568) = .6509$ . Now set  $N_1 = 81$ ,  $N_2 = 78$  and  $N_3 = 44$ . The values of  $\Delta A_i$  are:  $\Delta A_1 = .0226$ ,  $\Delta A_2 = .0224$ ,  $\Delta A_3 = .0321$  and the values of  $\Delta A_i / C(M_i)$  are:  $\Delta A_1 / C(M_1) = .0000545$ ,  $\Delta A_2 / C(M_2) = .0000543$  and  $\Delta A_3 / C(M_3) = .0001146$ . So, with  $N_3 = 44$  and  $N_1 = 80$  and with  $N_2 = 77$ , calculate A ( $A = .6753$ ). Since A is less than 0.85 it is necessary to perform other iterations until  $A \geq .85$ . The final

values for  $N_i$  are  $N_1 = 84$ ,  $N_2 = 80$ , and  $N_3 = 49$ . This process requires thirteen iterations and the final value of A is  $A = 0.8572$ .

The LCC for the system depends on the spares requirements that are necessary to meet the availability requirement. However, the minimization of the LCC is independent of the determination of the spares.

#### XVI. COMPUTER PROGRAM

A computer program has been written to accomplish the above described modularization. An explanation of the program and its flow chart are presented as Appendix 1. The program listing is presented as Appendix 2. Some output for the example problem is included as Appendix 3.

#### XVII. SUMMARY

This research has developed a method for modularizing an equipment subject to physical constraints, a mean-time-to-repair constraint and an availability constraint. The model for life cycle cost that was used is highly dependent on the physical configuration; namely, the number of external connections to a module.

Although the modular arrangements developed have not been proven to be optimal, no counter-examples have been seen. Further work would be desirable to 1) prove the method proposed is optimal, and 2) if 1) cannot be achieved, then develop an optimal mathematical solution to the problem.

The solution method and assumed model do not require a determination of spares requirements while performing the modularization process. It is recognized, though, that the cost of a module could be dependent on the total number of items purchased and in such a cases spares would enter



into the minimization process. The necessity for buffers between stages and a strong desire to group certain components into a given module were not addressed.

#### XVIII. REFERENCES

1. Jensen, P. A. "Optimum Network Partitioning", Operations Research, Vol. 19, No. 4 (July-August 1971) pages 916-932.
2. Jensen, P. A. A Graph Decomposition Technique for Design of Reliable Redundant Electronic Networks; unpublished Ph.D. dissertation, Johns-Hopkins University, Baltimore, Md. (September 1967).
3. Caponecchi, A.J. and Jensen, P.A. "A Partitioning Technique for Obtaining Solutions to the Modularization Problem", presented at the 40th National ORSA Conference, Anaheim, Calif. (October 1971).
4. Caponecchi, A. J. A Methodology For Obtaining Solutions to the Modular Design Problem, unpublished Ph.D. dissertation, The University of Texas at Austin, Austin, Texas (December 1971).

## APPENDIX I: Description of the Computer Program

The network partitioning program resides in the computer in the Multics environment. The program including its subroutines and the network data are stored in segments. This enables the user to access any part of the program for editing or compilation by calling the subroutine by name.

The names for the segments that contain the program parts are:

- 1) netpart.fortran
- 2) minx.fortran
- 3) sortx.fortran
- 4) sortxl.fortran
- 5) mshift.fortran
- 6) rshift.fortran
- 7) network.data

A brief description of the subroutines follows:

- 1) netpart--contains the coding for the algorithm described in Section XII (pages 15 and 16). (The program currently residing in the RADC computer can partition a network that has as many as eighty nodes and as many as ten constraints.)
- 2) minx--contains a subroutine called by netpart that selects the group of elements with the minimum value for the external minus the internal connections at each iteration.
- 3) sortx and sortxl--contain subroutines to properly arrange the identification of the elements in each module.
- 4) mshift--contains a subroutine to rearrange the data matrix so that every node is considered as a starting node for a set of modules.
- 5) rshift--contains a subroutine to reassign physical property values after the matrix has been rearranged by mshift.
- 6) network.data--contains all of the data relative to a given network.

It is assumed that the user can create segments where the data on the network to be partitioned can be stored. Data for different networks can be stored in different segments with different names. The data for the example network is stored in a segment called 'network.data'. When a user wants to store data it is necessary to enter:

'edm. "network data name" '

If the segment name already exists, the user will be put into the edit mode. If a new segment is being defined, the user will be put into the input mode and can enter the data on the new network. Fig. I-1 shows the procedure for creating a data segment. An explanation follows:

Line 1-- standard login procedure.

Line 10--user calls an already existing segment so Multics puts the system into the edit mode.

Line 11--'Q' is the quit command. No change is made in the data and Multics acknowledges by giving the system status data.

Line 13--user calls the edm function with a new name 'edm.ex.data' Multics searches for such a segment and since it doesn't find such a segment, responds with the message on line 14.

Line 15--indicates to the user to input his data.

Line 16--type '.' to change from input to edit mode after data has been entered.

Line 17--Multics response to the request to enter the edit mode.

Line 18--type 'Q' to quit without storing data, otherwise type 'W' with a carriage return and 'Q' to store new data.

The printout of the data for the example network is shown in Fig I-2. The input data should be put in in this same format. An explanation of Fig. I-2 follows.

Line 1-- number of nodes in the network, N.

Line 2-- number of physical constraints on the partitioning, k.

Lines 3,4&5--vectors  $CI$  where  $I=1,2,\dots,k$ . Each vector has N elements and there are k vectors.

Lines 6 through 18--the number of connections between nodes or elements.

Line 19--a vector that contains the limits on the vectors,  $CI$ .



After the user has entered the data for the network and attached it to the main program, the program is ready to be compiled and run. The output gives the best modular arrangement starting from each of the N nodes. The output also includes the value of ST (external minus internal connections) for each module. The user can then compare the N arrangements to determine the 'best' configuration. The output for a sample problem is shown in Appendix III.

```

1 Multics MR3.1X: KADC/Multics
  Load = 5.8 out of 21.5 units; users = 10
  Password:
  #####ected from preemption until 2030.
  Riesel 4540c0526 loaded in 09/30/76 1929.9 edt Thu from ASCII terminal "none".
  Last login 09/30/76 1926.5 edt Thu from ASCII terminal "none".
  No mail.
  locall: Imroser mode specification for this device. user_1/0
  r 1930 2.741 26.738 199

```

```

10 edm network.data
  Edit.

```

```

11 a
  r 1930 0.296 0.258 14

```

```

13 edm ex.data
14 Segment not found.
15 Input.
16 .
17 Edit.
18 a
  r 1931 0.117 0.000 0

```

Fig. I-1. Creating a Data Segment

## 09/27/76 1713.9 edt Mon

[illegible]

1714-0.315 1.364 47 Level 2, 12

105011

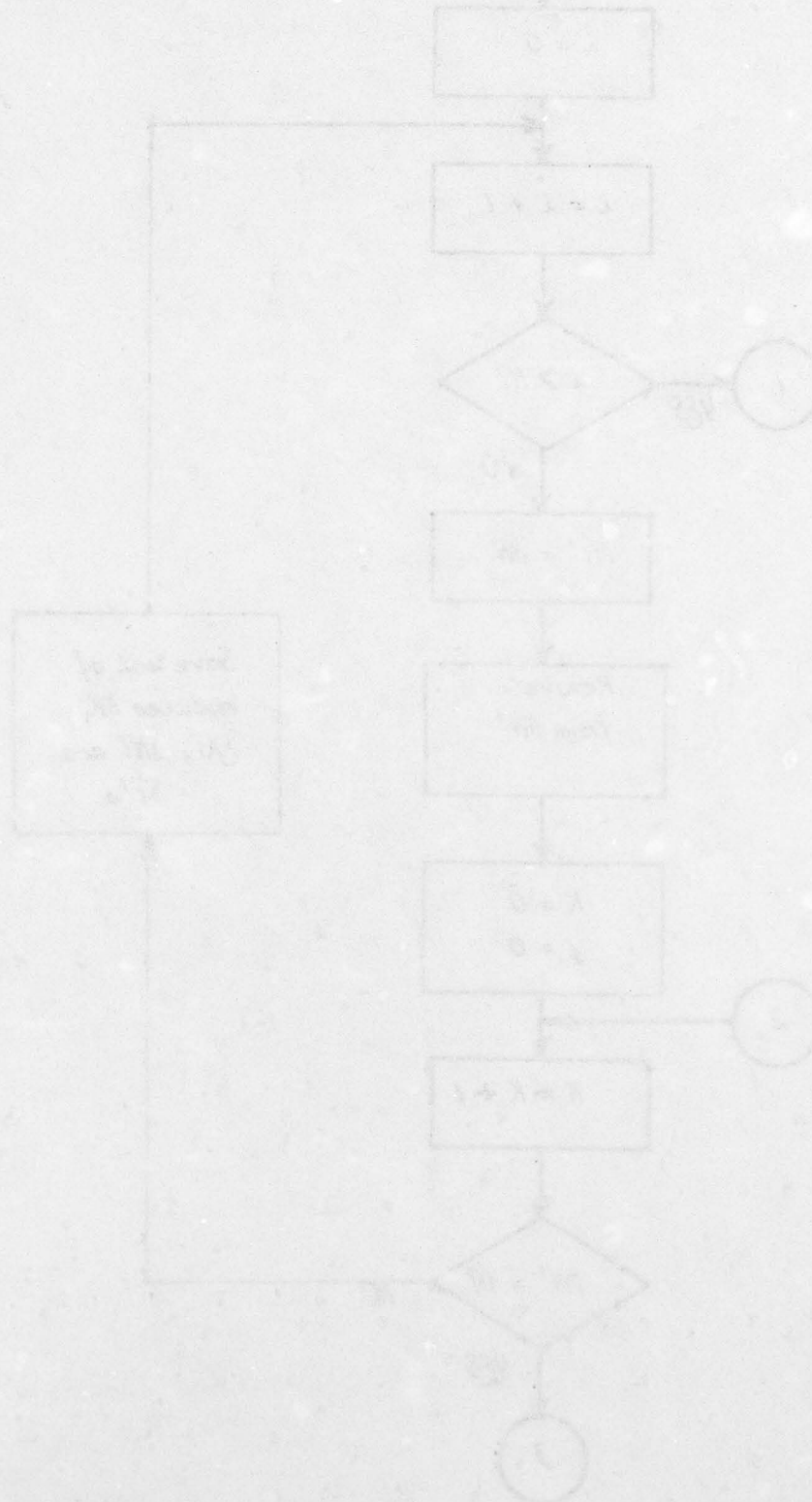
Piesol 4540C0526 logged out 09/27/76 1715.4 EDT Mon  
 CPU usage 5 sec, memory usage 40.9 units,  
 hangup.

Fig. 1-2: Printout of a Data Segment



## APPENDIX II: Flow Chart for the Computer Program

Appendix II contains the flow chart for the computer program previously described. The flow chart is presented in Fig. II-1.



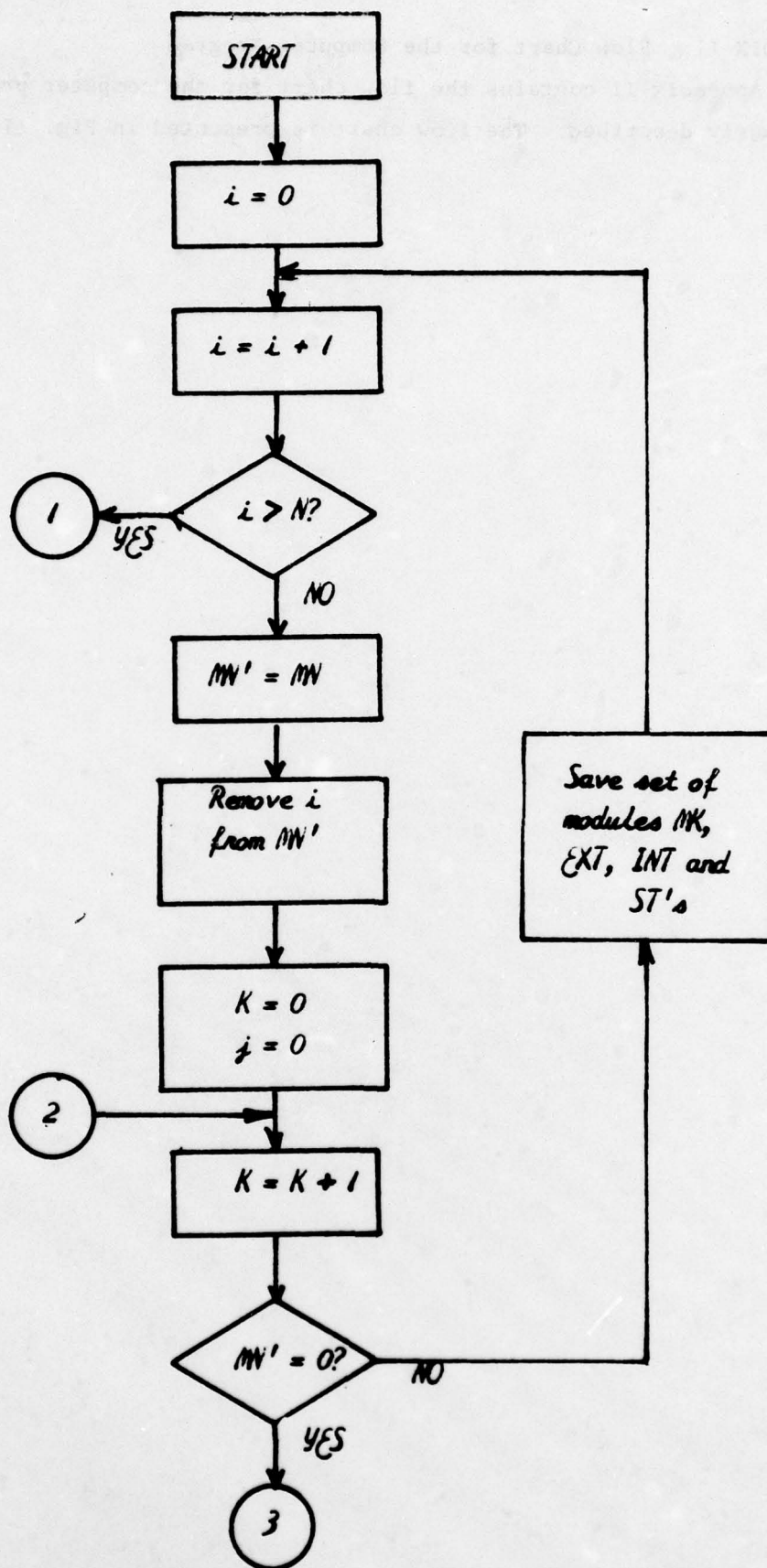


Fig. II-1: Flow Chart for Computer Program

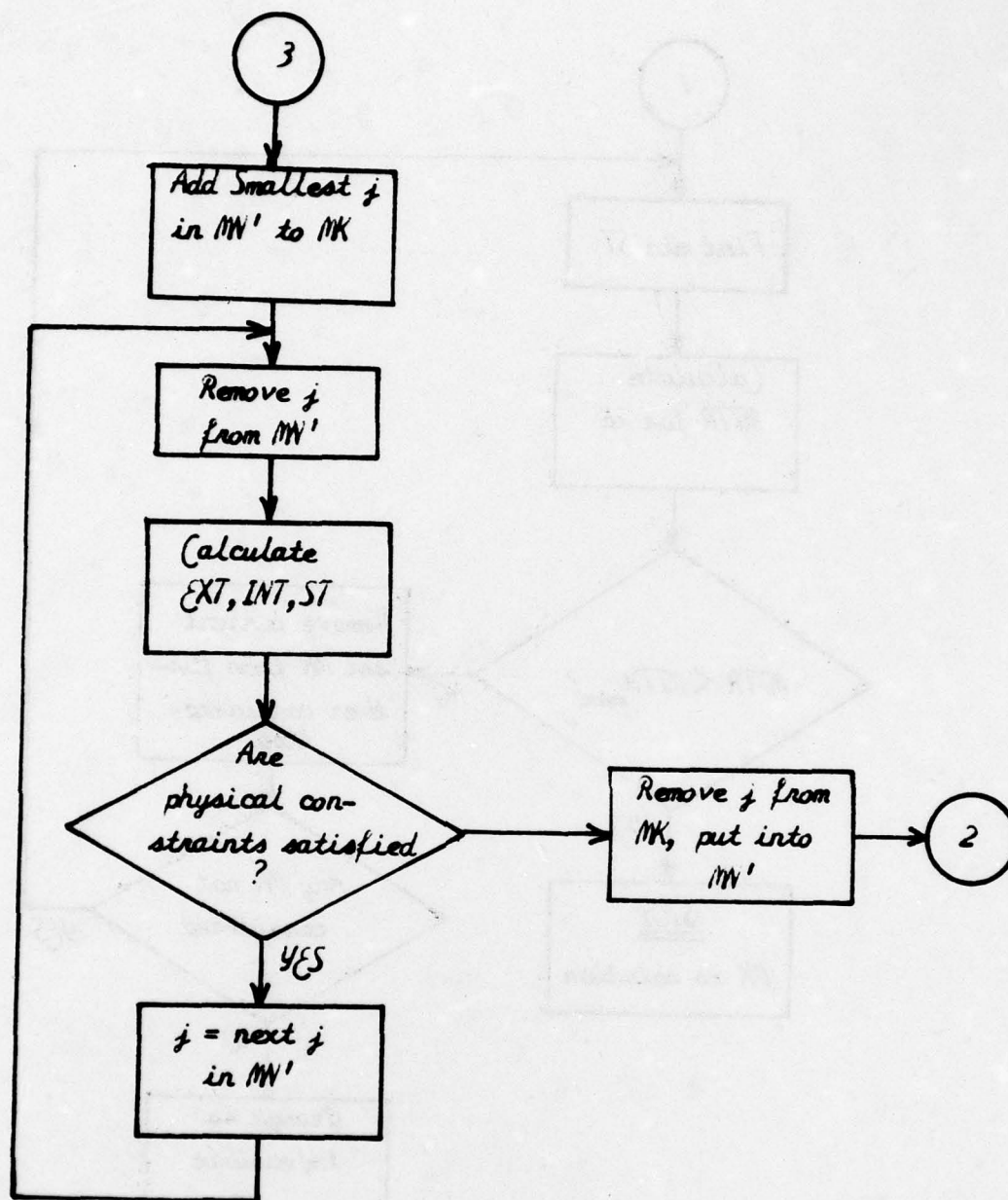


Fig. II-1 (cont)



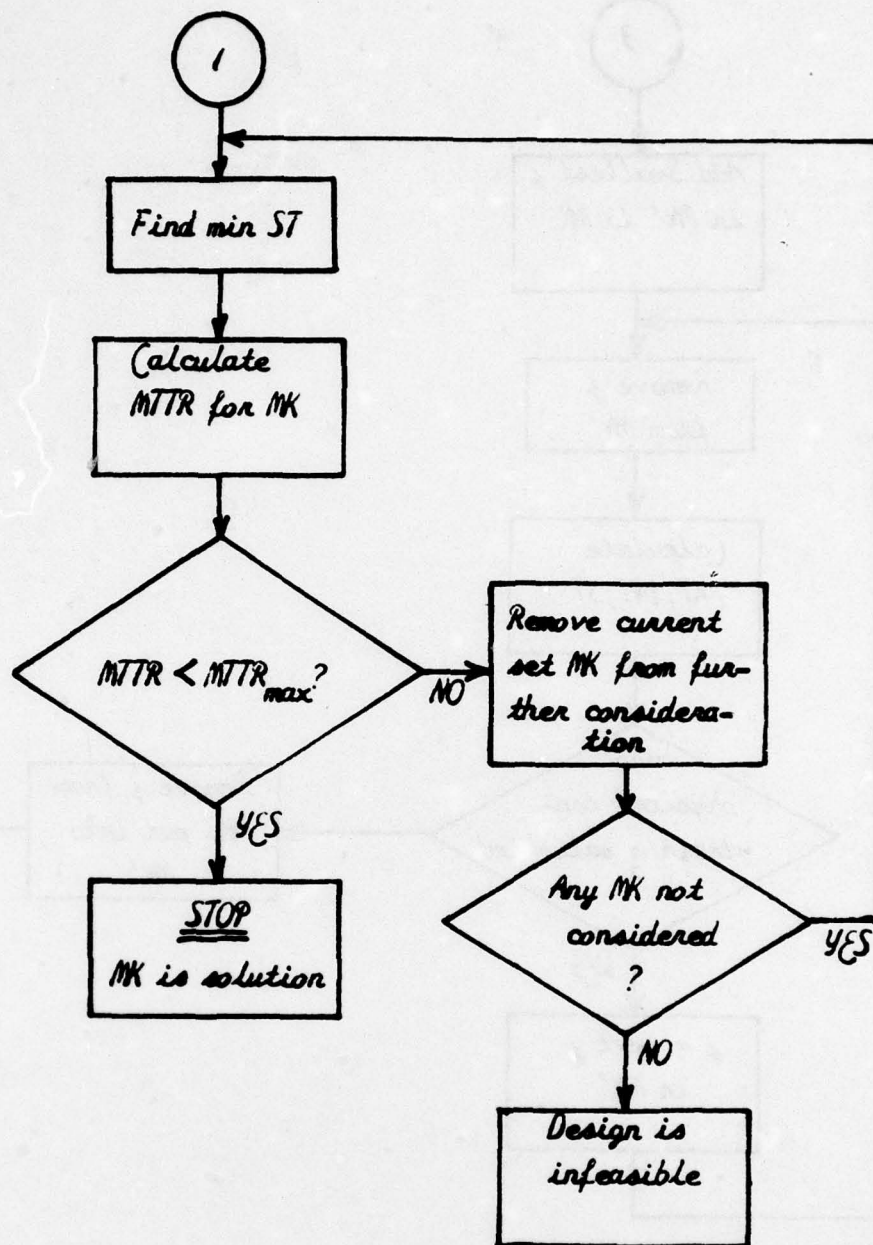


Fig. II-1(cont)

### APPENDIX III: An Example of the Output of the Computer Program

The output from the computer program for the example problem is shown in Fig. III-1. An explanation of Fig. III-1 follows.

Lines 1-9--standard login procedure

Line 10--asks the Fortran compiler to compile the main program segment containing netpart.

Line 13--the user asks the compilation of all of the subroutines by typing the name of each segment. Lines 16,19, 22 and 25 compile all of the segments.

Line 28--the input/output command which attaches the data to the main program.

Line 30--requests output.

Line 32--starts the computation.

**1-** Multics MR3.1X: RADC/Multics  
 Load = 16.3 out of 21.5 units: users = 21  
 Password:  
 [REDACTED]ected from preemption until 1537.  
 Biesel 4540c0526 lossed in 09/27/76 1437.2 edt Mon from ASCII terminal "none".  
 Last login 09/27/76 1355.3 edt Mon from ASCII terminal "none".  
 No mail.  
 iocall: Improper mode specification for this device. user\_i/o  
 r 1437 2.813 25.276 196

**10** ft netpart  
 Fortran  
 r 1438 8.958 34.328 284

**13** ft sortx  
 Fortran  
 r 1438 0.911 13.920 115

**16** ft sortx1  
 Fortran  
 r 1438 0.780 0.574 15

**19** ft mshift  
 Fortran  
 r 1438 1.313 24.806 165

**22** ft rshift  
 Fortran  
 r 1439 0.652 17.822 133

**25** ft minx  
 Fortran  
 r 1439 0.688 28.392 168

**28** io attach file01 vfile\_ network.data  
 r 1440 0.192 1.820 39



30 ioattach@io attach file03 swm\_user\_i/o  
 r 1441 0.060 0.678 25

32 netpart

1 network partitioning program

input matrix should be 13 by 13  
 0 starting node 1  
 0 card no. 1 contains nodes:  
 0 1 2 3 4  
 0

ext. conn. = 15  
 int. conn. = 12  
 st = 3  
 0 card no. 2 contains nodes:  
 0 5 6 9 10 12 13  
 0

ext. conn. = 14  
 int. conn. = 29  
 st = -15  
 0 card no. 3 contains nodes:  
 0 7 8 11  
 0

ext. conn. = 19  
 int. conn. = 6  
 st = 13  
 0 \* \* \* \* \* run completed \* \* \* \* \*  
 0 starting node 2  
 0 card no. 1 contains nodes:  
 0 2 3 4 8  
 0

ext. conn. = 2  
 int. conn. = 17  
 st = -15  
 0 card no. 2 contains nodes:  
 0 1 5 9 10 13  
 0

ext. conn. = 9  
 int. conn. = 10  
 st = -1  
 0 card no. 3 contains nodes:  
 0 6 7 11 12  
 0

ext. conn. = 34  
 int. conn. = 6  
 st = 28  
 0 \* \* \* \* \* run completed \* \* \* \* \*

```

0 starting node 3
0 card no. 1 contains nodes:
0 3 4 5 6 7
0

ext. conn. = -2
int. conn. = 21
st = -23
0 card no. 2 contains nodes:
0 1 2 7 11 12
0

ext. conn. = 9
int. conn. = 21
st = -12
0 card no. 3 contains nodes:
0 8 10 13
0

ext. conn. = 20
int. conn. = 0
st = 20
0 *****run completed*****
0 starting node 4
0 card no. 1 contains nodes:
0 4 5 6 10
0

ext. conn. = 2
int. conn. = 17
st = -15
0 card no. 2 contains nodes:
0 2 3 7 11 12
0

ext. conn. = 9
int. conn. = 10
st = -1
0 card no. 3 contains nodes:
0 1 8 9 13
0

ext. conn. = 34
int. conn. = 6
st = 28
0 *****run completed*****
0 starting node 5
0 card no. 1 contains nodes:
0 1 5 6 7 8 11
0

```

```

ext. conn. =      6
int. conn. =     27
st =    -21
0 card no.    2 contains nodes:
0      3      4      9      12      13
0

```

```

ext. conn. =     15
int. conn. =     13
st =      2
0 card no.    3 contains nodes:
0      2      10
0

```

```

ext. conn. =     29
int. conn. =      0
st =     29
0 * * * * * r u n c o m p l e t e d * * * * *
0 starting node    6
0 card no.    1 contains nodes:
0      6      7      8      9      12      13
0

```

```

ext. conn. =     -3
int. conn. =     22
st =    -25
0 card no.    2 contains nodes:
0      1      4      5      10
0

```

```

ext. conn. =     12
int. conn. =     13
st =     -1
0 card no.    3 contains nodes:
0      2      3      11
0

```

```

ext. conn. =     18
int. conn. =     13
st =      5
0 * * * * * r u n c o m p l e t e d * * * * *
0 starting node    7
0 card no.    1 contains nodes:
0      7      8      9      10      13
0

```

```

ext. conn. =     -2
int. conn. =     21
st =    -23
0 card no.    2 contains nodes:
0      2      3      5      6      11
0

```

```

ext. conn. =      9

```



```

int. conn. = 21
st = -12
0 card no. 3 contains nodes:
0 1 4 12
0

ext. conn. = 20
int. conn. = 0
st = 20
0 ***** run completed *****
0 starting node 3
0 card no. 1 contains nodes:
0 1 8 9 10 11
0

ext. conn. = -2
int. conn. = 21
st = -23
0 card no. 2 contains nodes:
0 3 4 6 7 12
0

ext. conn. = 9
int. conn. = 21
st = -12
0 card no. 3 contains nodes:
0 2 5 13
0

ext. conn. = 20
int. conn. = 0
st = 20
0 ***** run completed *****
0 starting node 9
0 card no. 1 contains nodes:
0 2 9 10 11
0

ext. conn. = 2
int. conn. = 17
st = -15
0 card no. 2 contains nodes:
0 3 6 7 8 12
0

ext. conn. = 22
int. conn. = 7
st = 15
0 card no. 3 contains nodes:
0 1 4 5 13
0

ext. conn. = 28
int. conn. = 10
st = 18
0 ***** run completed *****

```

```

0 starting node 10
0 card no. 1 contains nodes:
0 7 10 11 12
0

```

```

ext. conn. = 0
int. conn. = 19
st = -19
0 card no. 2 contains nodes:
0 4 5 8 9 13
0

```

```

ext. conn. = 10
int. conn. = 10
st = 0
0 card no. 3 contains nodes:
0 1 2 3 6
0

```

```

ext. conn. = 29
int. conn. = 18
st = 11
0 ***** run completed *****
0 starting node 11
0 card no. 1 contains nodes:
0 1 4 7 11 12 13
0

```

```

ext. conn. = 0
int. conn. = 27
st = -27
0 card no. 2 contains nodes:
0 2 6 9 10
0

```

```

ext. conn. = 11
int. conn. = 13
st = -2
0 card no. 3 contains nodes:
0 3 5 8
0

```

```

ext. conn. = 20
int. conn. = 0
st = 20
0 ***** run completed *****
0 starting node 12
0 card no. 1 contains nodes:
0 1 2 5 8 12 13
0

```

```

ext. conn. = 0
int. conn. = 27
st = -27
0 card no. 2 contains nodes:
0 3 6 7 10 11
0

```

```

ext. conn. = 12
int. conn. = 13
st = -1
0 card no. 3 contains nodes:
0 4 9
0

```

```

ext. conn. = 19
int. conn. = 0
st = 19
0 * * * * run completed * * * *
0 starting node 13
0 card no. 1 contains nodes:
0 1 2 3 6 13
0

```

```

ext. conn. = -2
int. conn. = 21
st = -23
0 card no. 2 contains nodes:
0 4 8 9 11 12
0

```

```

ext. conn. = 9
int. conn. = 21
st = -12
0 card no. 3 contains nodes:
0 5 7 10
0

```

```

ext. conn. = 20
int. conn. = 0
st = 20
0 * * * * run completed * * * *
STOP

```

```

fortran_io_: Close files? logout
Please answer "yes" or "no". yes
r 1446 3,919 18,007 336

```

logout



# METRIC SYSTEM

## BASE UNITS:

Quantity	Unit	SI Symbol	Formula
length	metre	m	...
mass	kilogram	kg	...
time	second	s	...
electric current	ampere	A	...
thermodynamic temperature	kelvin	K	...
amount of substance	mole	mol	...
luminous intensity	candela	cd	...

## SUPPLEMENTARY UNITS:

plane angle	radian	rad	...
solid angle	steradian	sr	...

## DERIVED UNITS:

Acceleration	metre per second squared	...	m/s
activity (of a radioactive source)	disintegration per second	...	(disintegration)/s
angular acceleration	radian per second squared	...	rad/s
angular velocity	radian per second	...	rad/s
area	square metre	...	m
density	kilogram per cubic metre	...	kg/m
electric capacitance	farad	F	A·s/V
electrical conductance	siemens	S	A/V
electric field strength	volt per metre	...	V/m
electric inductance	henry	H	V·s/A
electric potential difference	volt	V	W/A
electric resistance	ohm	...	V/A
electromotive force	volt	V	W/A
energy	joule	J	N·m
entropy	joule per kelvin	...	J/K
force	newton	N	kg·m/s
frequency	hertz	Hz	(cycle)/s
illuminance	lux	lx	lm/m
luminance	candela per square metre	...	cd/m
luminous flux	lumen	lm	cd·sr
magnetic field strength	ampere per metre	...	A/m
magnetic flux	weber	Wb	V·s
magnetic flux density	tesla	T	Wb/m
magnetomotive force	ampere	A	...
power	watt	W	J/s
pressure	pascal	Pa	N/m
quantity of electricity	coulomb	C	A·s
quantity of heat	joule	J	N·m
radiant intensity	watt per steradian	...	W/sr
specific heat	joule per kilogram-kelvin	...	J/kg·K
stress	pascal	Pa	N/m
thermal conductivity	watt per metre-kelvin	...	W/m·K
velocity	metre per second	...	m/s
viscosity, dynamic	pascal-second	...	Pa·s
viscosity, kinematic	square metre per second	...	m/s
voltage	volt	V	W/A
volume	cubic metre	...	m
wavenumber	reciprocal metre	...	(wave)/m
work	joule	J	N·m

## SI PREFIXES:

Multiplication Factors	Prefix	SI Symbol
1 000 000 000 000 = 10 <sup>12</sup>	tera	T
1 000 000 000 = 10 <sup>9</sup>	giga	G
1 000 000 = 10 <sup>6</sup>	mega	M
1 000 = 10 <sup>3</sup>	kilo	k
100 = 10 <sup>2</sup>	hecto*	h
10 = 10 <sup>1</sup>	deka*	da
0.1 = 10 <sup>-1</sup>	deci*	d
0.01 = 10 <sup>-2</sup>	centi*	c
0.001 = 10 <sup>-3</sup>	milli	m
0.000 001 = 10 <sup>-6</sup>	micro	μ
0.000 000 001 = 10 <sup>-9</sup>	nano	n
0.000 000 000 001 = 10 <sup>-12</sup>	pico	p
0.000 000 000 000 001 = 10 <sup>-15</sup>	femto	f
0.000 000 000 000 000 001 = 10 <sup>-18</sup>	atto	a

\* To be avoided where possible.

# *MISSION of Rome Air Development Center*

*RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C<sup>3</sup>) activities, and in the C<sup>3</sup> areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.*

